NASA-CR-203406

NAS 2-13721
711-61-CR
026185

A Framework for the Design of Effective Graphics for Scientific Visualization*

Kristina D. Miceli[†]

Report RNR-92-035, December 1992

NASA
National Aeronautics and Space Administration

Ames Research Center
Moffett Field, California 94035

ARC 275a (Feb 81)

A Framework for the Design of Effective Graphics for Scientific Visualization[*]

Kristina D. Miceli[†]
Report RNR-92-035, December 1992

Abstract

This proposal presents a visualization framework, based on a data model, that supports the production of effective graphics for scientific visualization. Visual representations are effective only if they augment comprehension of the increasing amounts of data being generated by modern computer simulations. These representations are created by taking into account the goals and capabilities of the scientist, the type of data to be displayed, and software and hardware considerations. This framework is embodied in an assistant-based visualization system to guide the scientist in the visualization process. This will improve the quality of the visualizations and decrease the time the scientist is required to spend in generating the visualizations. I intend to prove that such a framework will create a more productive environment for the analysis and interpretation of large, complex data sets.

[*]A doctoral dissertation proposal presented to the Department of Computer Science, University of Colorado, Boulder, November 20, 1992.

[†]Numerical Aerodynamic Simulation (NAS) Systems Division, MS T045-1, NASA Ames Research Center, Moffett Field, CA 94035-1000 (kmiceli@nas.nasa.gov). The author is an employee of Computer Sciences Corporation. This work is supported through NASA Contract NAS 2-12961.

1 Introduction

Computer simulations are producing increasingly larger data sets. To perform a thorough analysis of these data sets simply by studying the numbers is difficult, if not impossible. As a result, the technology to graphically display this data is essential. Scientific visualization is an enabling technology that assists the scientists with the analysis of the large data sets currently being produced.

A number of visualization systems exist to analyze simulation data sets [1, 21, 35, 39]. However, each of these systems requires the scientist to build each visualization explicitly. The scientist is required to know what types of representations are available and which representations fit which types of data. Scientists typically do not have a background in visualization, so they often seek assistance from a visualization expert to help them display their data. If such an expert is not available, it is up to the scientist to generate his visual representations. Not only does this require a great deal of time in learning to use the tools, but, because the scientist does not have a background in visualization, he could generate a representation that leads to confusion or misinterpretation. For these scientists, the complexity of current visualization systems has caused them to avoid using the technology that could potentially be very valuable to them.

For example, a fluid dynamicist is interested in viewing a velocity vector field computed over the wing of an airplane. The fluid dynamicist is required to 1) determine what visualization techniques are useful in viewing velocity fields, 2) find the software that implements such techniques, 3) learn to use the software so that she can create the visualization, and 4) correctly select from the many options to enhance the image such as color, lighting, viewpoints, etc. If the fluid dynamicist is interested in viewing a complex relationship in her data set, such as how the pressure of the fluid flow affects the deformation of the wing, the risk of developing poor representations is increased as is the amount of time spent using the software tools. This time could be better applied to analyzing the data and understanding its behavior.

The rapid generation of visualizations is especially important due to the exploratory nature of scientific domains, in which visualizations will be viewed briefly and then discarded. Scientists need a tool to support the data analysis process. This tool must give them easy access to visualization technology so they may concentrate on their scientific problem. Current visualization tools impede this process by requiring the scientist to know how to apply visualization technology to their domain.

To maximize the benefit of the vast amount of scientific information produced by computers, intelligent systems that can assist the scientist in performing productive visual data analysis would be beneficial. A computer visualization assistant would provide guidance to scientists who do not have visualization expertise nor access to a human visualization expert. Several benefits would arise by providing this assistance to the scientist. Since the scientist is often unaware of the constraints imposed by computer software, hardware and the human visual system, the resulting visualizations will be better suited for the task at hand. The amount of time learning how to use visualization software and generating visualizations will be reduced, allowing the scientist to concentrate his time on studying the simulation results.

The development of an assistant-based system entails several components. Information about the data, the user, and the available hardware and software is important in order to generate the appropriate visualization. In addition, knowledge from graphic design, visual perception, and the scientific domains provides essential information. The information from these sources can be used to produce visualizations based on the goals and capabilities of the scientist, the type of data to be displayed, and software and hardware resources.

The integration of this information forms a complex system. In order to abstract this complexity, the development of models to describe each component of the visualization system is helpful. Data modeling involves organizing the data into an appropriate form so that information can be extracted easily. User modeling describes the goals and characteristics of the user so the system can better adapt to her requirements and preferences. The machine model is necessary to determine how each aspect of the computer, both software and hardware, impacts the resulting visualization and interaction technique. Tying together the information in these models is the knowledge base, containing facts and rules from graphical presentation, visual perception and expertise in the scientific domains. The information in the knowledge base can be coupled with the data, user and machine models, presenting the scientist with effective visualizations.

The development of a visualization system requires the integration of the components mentioned above in order to provide the scientist with a comprehensive system for the production of effective visual representations. This proposal presents work in building the base of this visualization system to assist the scientist with the visualization process. An overview of the proposed research project is presented in Section 2. Section 3 reviews past and current research that has impacted this proposal. A more thorough description of the research proposal is presented in Section 4. This section also includes a description of the specific application domain on which this model will be tested. Section 5 describes the contributions of this research and section 6 presents a timeline that describes how this project will be implemented. Finally, in Section 7, conclusions as well as plans for additional research are presented.

2 Research Proposal - An Overview

The goal of this project is to assist the scientist with the design of effective graphics for scientific visualization. Effective graphics, in a general sense, describe useful visual representations that aid the scientist with the interpretation of data. Mackinlay[22] uses the term "effective", along with the term "expressive", more specifically in characterizing what exactly defines a useful visualization. In his work, "effective" suggests how well a graphical technique exploits the capabilities of the output medium and human visual system to ensure a correct and quick judgement of the data. The complementary word "expressive" describes how well a graphical representation encodes the data attributes, presenting all the relevant information (and *only* the relevant information) to the user. The use of these terms have become somewhat standard, used by many as the criteria in designing graphical presentations. These terms are used in this proposal for evaluating visual representations of scientific data.

The goal of this project manifests itself in two ways. First, effective visualizations will be produced that will aid the scientist in interpretation by taking into account rules of graphic design, perception and knowledge in the scientific domains. Second, visual data analysis will be more productive for the scientist, because the scientist will no longer be required to understand visualization techniques and visualization systems. Instead, they will be provided with a system that incorporates this knowledge, allowing them to spend the maximum amount of time analyzing their data with the minimum amount of effort.

In order to achieve this goal, an assistant-based system will act as a visualization expert in guiding the scientist in the creation of visualizations. This system will be based on a framework that consists of a data model, a user model, and a machine model. The integrating component of the framework will be the knowledge base. The system uses information stored in the knowledge base, together with the information in the individual models to suggest useful visualization techniques.

The data model represents the data contained in the scientific domain. The data for each domain contains the geometry of the simulated object, the data variables that have been calculated in the simulation, and the relationships between the data variables. This information is embodied in an object-oriented model.

To complement the data model, the user model describes each scientist and her characteristics. These characteristics include the interpretation aims of the scientist, the background of the scientist, and any limitations the scientist may possess. The interpretation aims specify what information the scientist is trying to extract from the data. This information can be detailed (i.e. locating certain values of data points) or the information can be general (i.e. investigating local trends in the data). Background information about the scientist might include information such as what type of computer she uses or what type of user interface she prefers. The limitations of a scientist are intended to describe physical constraints such as a color blindness.

The machine model incorporates information about the resources available in the scientist's computing environment. This includes information about the characteristics and limitations of the software and hardware available for performing visual data analysis.

The knowledge base contains information that influences the mapping of data to visualizations. The scientist is often unaware of the constraints imposed by the computer and the human visual system. The information in the knowledge base, obtained from the fields of graphical presentation, visual perception and from the individual domains, attempts to fill this void.

This framework defines the components that are critical in the visualization process. However, each of the components in this framework is a research project in itself. Data models, user models and knowledge base systems have been studied for many years in the database and artificial intelligence fields and research is still ongoing in these areas. As a result, a simplified version of this framework will be examined in order to thoroughly study the principal component of the framework, the data model, and how the mapping of data to visualizations is performed using rules in the knowledge base.

The data model was selected as the main and driving component of the visualization framework. The characteristics of the data define what visualization technique should be applied to the data. The information in the user and machine models is important in the selection of this mapping. However, the user and machine models only support and refine the mappings from data to representation. Their impact can be studied as a later research project in the development of this visualization framework.

The data model will characterize data sets involved in a multidisciplinary (fluid dynamics and structural dynamics) project currently in progress at the NASA Ames Research Center. The mappings from data to visualization will be based on simplified user and machine models. The user model will characterize a computational scientist involved in the project, his preferences and working environment. The hardware model will describe the system that is in the scientist's working environment at NASA Ames. The knowledge base will contain information about visual presentation techniques, perceptual considerations and knowledge from the scientific domains.

The end product of this research project will be a prototype version of an assistant-based visualization system. With this prototype system, the scientist will be able to select data objects and express his interpretation aim to the system. The system will process this information and suggest a set of visualizations to the scientist, completely rendered and annotated. The scientist will be informed about the properties of the visualizations and why they were selected. The scientist may override these selections at any time.

This prototype system will be implemented in Superglue[16], an object-oriented programming environment based on the language Scheme. Superglue is under development at the NASA Ames Research Center and supports a rich base of visualization primitives. As a result, the programming of visualization techniques can be kept to a minimum by taking advantage of previously written software.

In order to show that an assistant-based system of this type is advantageous in analyzing the large data sets, scientists will be interviewed during the project phases and upon completion of the project. Questions will be asked regarding the effectiveness of this type of approach in analyzing data. Interaction with the user will occur throughout the development of the system. This will provide valuable input to the direction of the project and will influence the design of the user interface.

3 Related Work

This project presents the development of a data model as the foundation of an assistant-based visualization system. The use of models is a successful method for the design and implementation of complex systems. Model building allows for a disciplined approach to the design process by adhering to principles of decomposition, abstraction and hierarchy[3]. The following sections describe the use of data models and visualization models in previous research.

3.1 Data Models

Data models have been used successfully in database applications to organize large amounts of data. Different modeling paradigms have evolved as the needs for data representation have changed. The selection of one of these modeling paradigms is necessary to implement the data model for this project. It is important to select the appropriate paradigm, since this defines the structure for the data representation. The following paragraphs describe the different paradigms used for data modeling: the hierarchical model, the network model, the relational model, the semantic model and the object-oriented model.

The hierarchical data model is the oldest type of data model. It is based on the concept that data in the real world can be perceived and organized in a hierarchical manner. These relationships are captured in hierarchical tree structures that are ordered from parent to child. Each node of this tree is an entity, called a record type, that is composed of one or more attributes, called data items, that describe the entity[19].

The network model was based on the 1971 report published by the CODASYL Data Base Task Group[25]. The network model is composed of entities that are defined by record-type definitions. These definitions consist of a name and the fields that describe the entity being specified. Relationships between entities are specified by set types, where a set is composed of an owner record type and a member record type. This structure allows for a one-to-many mapping and complex structures are formed when sets are allowed to intersect.

The relational model was popularized in the early 1970s, replacing the hierarchical and network models as the premier data model in research and commercial applications. Structurally, the user sees the relational database as a collection of tables called relations. The rows of a table are called tuples and represent instances of a entity. The columns of the relation are the attributes of the entity type. The domain is the set of all values that can appear in a given column. Relational data models are operated on using the relational algebra. The relational algebra consists of operators that perform tasks such as union, intersection, difference, select, project and join. Each of these operators takes one or more operands and produces a new relation as a result[19].

During the time of interest in relational models, the semantic model[14] was developed due to the lack of semantics in the hierarchical, network and relational models. The semantic model provides a more natural way to specify the design of a database. The model allows for the representation of objects of interest in a way that more closely resembles the view the user has of these objects and relationships. These abstractions allow the designer to model an abstract object based on the properties or attributes of the object. The capability to derive data is another appealing aspect of the semantic model. This capability allows the designer to have access to data values that do not necessarily need to be stored, but which can be computed as needed.

Object-oriented models rose to popularity from their predecessor, the semantic model. The object-oriented model is superior to its predecessors in its ability to model both the structural and behavioral components of data. As a result, object-oriented models represent entities the way they are perceived in the the real world. Object-oriented models have support for general data types, nested objects, and allow for compute-intensive applications.

Object-oriented models are superior to hierarchical, network, relational and semantic models when dealing with scientific data because they offer support for the complexities inherent in scientific data. Traditional models do not fit the needs of the multidimensional, heterogeneous, hierarchical structures found in scientific data. Concepts such as a class hierarchy, inheritance, and methods are not present in the hierarchical, network and relational data models. Semantic models lack the methods that allow for the manipulation of the behavioral state of the model. Therefore, the object-oriented modeling paradigm is the most appropriate model for managing the complexity of scientific data.

The main concepts of the object-oriented paradigm include abstraction, encapsulation, and inheritance[3]. Abstraction is an effective way to deal with complexity, since it breaks detailed systems into simple conceptual objects with distinct boundaries. Abstraction is possible due the similarities between objects and processes in the real world. It takes advantage of these similarities, providing groupings so that the complexities are hidden and only the basic concepts are obvious to the viewer.

Encapsulation captures the essence of the object paradigm, providing a clear separation between the external workings of objects and their internal implementation. Encapsulation and abstraction are complimentary to each other. Abstraction focuses on the outside view of the object, while encapsulation provides a mechanism for hiding the inside, detailed view, which does not need to be seen by the user.

Although abstraction allows for the decomposition of complex systems, there is usually more information than the user can handle. The need arises for some type of ordering mechanism to control these abstractions so that the user may comprehend the overall system. Inheritance is one mechanism that allows for this ordering. Inheritance defines relationships among objects. A class of objects can inherit structure or behavior from another class of objects.

Object-oriented data models have the capability to represent and manipulate complex, nested objects. These models also provide rich mechanisms for representing structurally complex interrelations among scientific data. Included in the benefits of the object-oriented paradigm are the increased separation of conceptual and physical components, decreased semantic overloading of relationship types, and the availability of convenient abstraction mechanisms.

3.1.1 Data Models in Visualization Systems

For the most part, visualization systems ignore the issue of data management and data representation. Typically, access is provided via a flat file structure and the format of the data is hidden within the software. Data is not dealt with coherently and data formats often limit the size and dimensionality of data sets and the knowledge about the data that can be represented.

Designers of several visualization systems have begun to acknowledge the need for data models in the form of flexible data structures and standard data formats. However, most of these systems place the emphasis on data-structure-driven models instead of knowledge- (or domain-) driven models [37]. The result of this emphasis is the loss of the semantics contained in the data. This is the main difference in the models presented below and the data model proposed in this project.

Many of the popular commercial systems use the data flow paradigm. The data in these types of systems are stored as a stream of bytes, transferred between the modules that comprise a data-flow "map". Typically, a data set is read from a file using one module, passed as a stream of bytes to another module where it is transformed into a user-selected visualization, and then passed to the rendering module for display.

The Explorer package from Silicon Graphics employs the data-flow paradigm. Explorer has a flexible data model (structure) that describes the complex data sets that currently exist. Although this data structure is very accommodating, data still exist in large files. When the modules in the Explorer map are executed, the large data sets simply flow through each module in the map.

IBM's Data Explorer is also a data-flow system. This data model is similar to Explorer's in the sense that it is structure based rather than knowledge- (or domain-) based. Structures that can be represented include data defined on a regular orthogonal grid, data on a deformed regular grid and data on a variety of irregular grids such as triangular, quadrilateral and tetrahedral grids. These underlying data structures are important, but the semantics of the data set are lost and cannot be used.

Other examples of data models include formats currently used for storing and transferring data. Again, these examples are more data formats based on flexible data structures than data models that represent semantic knowledge about the data sets. NASA's CDF (Common Data Format) was one of first implementations of a data model[36]. It is based on the concept of providing abstract support for the class of data that can be described by a multidimensional block structure. Unidata's netCDF is another model focused on issues of data transport[26]. HDF (Hierarchical Data Format), from NCSA, concentrates on the need to move files of data among heterogeneous machines[24].

3.2 Visualization Models

Models and systems for the automatic generation of effective graphical presentations have recently developed [4, 9, 10, 22, 28, 29, 30, 31, 32, 33, 41]. The applications vary in each of these research projects, from the presentation of charts and graphs to the presentation of images. However, they share a common goal of developing tools and techniques to make the data analysis process more intuitive and productive. Researchers in this area have recognized the importance of the several disciplines necessary for the creation of effective graphical presentations, namely computer graphics, data management, graphic design, perceptual psychology and user interface design. All researchers strive to define the components of the graphic design process, viewing current *ad hoc* methods as unacceptable for future graphics and visualization systems.

Visualization is essentially a mapping process, taking data attributes and mapping them into visualizations[7]. Data analysis systems are emerging that perform this mapping based on rules of perception and graphical presentation. Feiner[10] calls these systems "graphically articulate" in that they attempt to accurately convey the meaning of the representation to the user. These systems can have varying levels of autonomy in the user interface, to suit the needs and capabilities of the scientist. A computerized assistant can suggest appropriate representations based on the characteristics of the data. A completely autonomous system can create representations and identify sources of interest based on mathematical analysis, pattern matching and a large database of information obtained from the user.

The issue of automated versus semi-automated systems is one of importance. Completely autonomous systems are typically not successful due to the vast amount of knowledge and inference capabilities they must contain. Semi-automated systems, therefore, are probably more likely for visualization systems in the near future. Paradigms for semi-automated systems include assistant-based, critic-based, improver-based and cooperative computer-aided design[10]. Assistant-based systems offer suggestions to the user based on their specific goal and the information in the knowledge base. Critic-based systems critique the visualization created by the user based on the information in the knowledge base. Improver-based systems add or enhance the design created by the user automatically. Finally, cooperative, computer-aided designs involve the expertise of both the user and the computer, taking turns in developing representations in an iterative manner.

Some of the research performed in this area is presented below. The systems have developed from conceptual ideas to simple graphics tools to visualization systems that help the user with the design of graphical presentations.

Haber and McNabb [13] developed the concept of "visualization idioms", a model to aid in the generation of complex visualizations of large scale numeric simulations. The visualization model is presented as a set of generalized mappings that transform raw data into geometrical abstractions. The goal is to convert the raw data into a format that can be understood by the human visual system while maintaining the integrity of the data. Three transformations occur in the process of transforming raw data into a visualization. The first transformation involves enriching or enhancing the data, in order to process it into a form fit for visualization. The next transformation involves the visualization mapping, or the creation of an "abstract visualization object" (AVO) that maps the simulation data into attribute fields. These attribute fields might include graphical qualities such as geometry, time, color, transparency, luminosity, etc. The final transformation of the data involves rendering. The resulting images, or "visualization idioms", refer to the abstract meaning of pictures as a visualization of scientific data sets.

Mackinlay[22] developed APT (A Presentation Tool) to automatically generate graphical presentations of relational information. The model is described by data, task and user directives. APT contains a description of the information and task, defines evaluation criteria in formal terms, and is able to compose multiple items and relations into one effective display. Mackinlay uses composition rules to define appropriate combinations of simple graphics primitives in the generation of representations. To map from an internal representation to displayable images, evaluation criteria such as importance (ranking of tasks), expressiveness (encoding of data attributes), and effectiveness (psychophysical principles) are used. Although restricted to relational data and two-dimensional charts, Mackinlay's work is a foundation for complex visualization systems aiming towards automating the design of graphical presentations. Problems and areas in which Mackinlay believes additional work is necessary include an automated presentation tool for three-dimensional data as well as tools for automating the extraction of features in the data, the interpretation of these features and the discovery of new types of phenomena in the data.

Robertson [28, 29, 30] developed visualization guidelines by observing natural scenes and matching data and task characteristics to two-dimensional and three-dimensional scene parameters. Robertson's approach is different from previous techniques in that he has a complete and coherent scene in mind already before the mapping stage begins. By restricting the user to predefined scenes, the natural scene paradigm guarantees perceptually valid mental models. De Ferrari [30] expanded this original paradigm to include a more elaborate data model, user directives, and user interpretation aims (the latter two combined under the term "visualization specification") as input to the visualization system. The goal of this model is the automatic generation of visualizations.

Wehrend and Lewis[41] describe each visualization process by two dimensions: *a)* the characteristics of the information to be displayed and *b)* the specific perceptual task to be performed on the resulting images. The finite number of data characteristics and the finite number of perceptual tasks define a two-dimensional matrix in which each element contains expressive and effective examples of visualizations. If more than one tuple *(data, task)* is to be represented in the same display, the user is responsible for setting priorities for the composition of representations.

The VISualization Tool Assistant (VISTA) [32] is a system, developed by Senay and Ignatius, which generates visual representations automatically. VISTA emphasizes the mapping of data attributes to primitive visualization techniques, which encode one dependent and up to four independent variables. The synthesis of visualization techniques takes place in three steps. The first step involves decomposing the data so that each element can be represented by a single visualization primitive technique. The second step involves using rules of effectiveness and expressiveness to find the proper visualization technique for each data component. VISTA uses a depth-first search until it finds a visualization technique that can express a given relation, provided that the visualization technique has not already been used to visualize another relation. In the final step, primitive techniques are combined to form a composite visualization by applying the appropriate composition rules. The user is able to interactively modify certain attributes of the visualization without causing inconsistencies in the final design.

4 A Framework for the Design of Effective Graphics for Scientific Visualization

This section describes the framework of an assistant-based system for the visualization of complex data sets, following the lead of the research mentioned in the previous section. Although the entire framework is presented below for completeness, this project and this proposal emphasize the data model as the central and driving component. The information in the knowledge base is also important in that it provides the facts and rules that establish the visualization mappings.

In order to perform a proof of concept, a specific application, multidisciplinary fluids/structures interaction, has been chosen to test the development of the visualization framework. The following sections describe the environment and data involved in the multidisciplinary simulations performed at the NASA Ames Research Center.

4.1 Multidisciplinary Visualization for Computational Aerosciences

Computational fluid dynamics (CFD), the study of fluid flow via numerical simulation, assists scientists and engineers in developing a better understanding of fluid flow and how it affects the flight characteristics of aircraft and aerospace vehicles. These simulations can involve large data sets composed of multiple grids and multiple physical variables. For example, a recent simulation of an F-18 aircraft[27] involved a grid composed of 1.6 million points, each with five physical variables per instant in time.

The data involved in these simulations typically include one or more computational grids filling a volume around the object, together with scalar and vector fields defined on these grids. Many additional variable fields can be derived from the existing data using the laws of physics. These variables describe the characteristics of the vehicle in the computer simulation. The structure of these data sets and their interrelations are important to describe so that the maximum amount of information may be obtained from the simulation results.

With increases in computing power, scientists have become interested in the study of integrated, multidisciplinary simulations. These simulations can involve interactions between disciplines such as fluid dynamics, structural dynamics, chemistry, combustion, and controls. Large, single discipline CFD data sets are complex and difficult to manage. The combination of data sources from multiple disciplines raises the importance of data management and representation. Methods and models to organize data and represent relationships between data variables and data sets are as important to visualization research as visualization techniques.

Multidisciplinary simulations require the use of both structured and unstructured grids. Structured grids are based on a rectangular connectivity that defines the nearest neighbor for each element in the grid. However, the actual physical locations of the grid points can form curved and warped surfaces in an attempt to closely model the surface of the geometry. Unstructured grids do not possess this same rectangular connectivity. The grid cells in an unstructured grid can be a variety of shapes (i.e. triangles, quadrilaterals, tetrahedra, prisms). The connectivity of these elements is explicitly specified by a list which contains the grid points in the order in which they are connected.

One of the multidisciplinary data sets that will be used to test this assistant-based visualization system is the High Speed Civil Transport (HSCT). The geometry of the HSCT in the structures domain is described by an unstructured grid. The grid consists of approximately 9,000 points and the simulation involves thousands of timesteps. The structures simulation calculates the deformation (vector) and stress (scalar) values at each point in the grid. The variables that describe the grid will include the physical-space coordinates (vertices) of the structure, the connectivity of these vertices, and two physical variables: deformation and stress.

The information the structural dynamicist is trying to extract from the data is how the object deforms and the resulting stress levels on the skin of the vehicle. Some of the visualization techniques that are helpful to the scientist include color maps and contour lines to show the stress levels and animation to show the deformation of the vehicle. Another visualization technique that is helpful to the scientist is a plot that shows deformation over time. This information is useful in determining when "flutter" begins and how it progresses. The ability to narrow the view to a region of interest is of importance to the scientist, as are standard functions such as changing the point of view, zooming, controlling the color range, and positioning and viewing cutting planes.

The multidisciplinary simulation will help fluid dynamicists to understand how the deforming body of the HSCT affects flow characteristics. The fluids simulation will involve approximately 79,000 points that define the fluid volume around the HSCT. The fluids data set lies on multiple grids that describe the volume about the HSCT. The physical variables include one vector field, momentum, and two scalar fields, energy and density. Many additional variables can be calculated from these variables, including the velocity and pressure of the flow.

Many visualization techniques exist for viewing fluid flow. Standard visualization techniques such as the use of color, contours, and plots provide essential functionality. Visualization techniques such as particle traces and stream surfaces are useful for visualizing the velocity field of the fluid[15]. Cutting planes are effective for viewing slices of data, such as the cross section of a wing. Vector field topology[11] and volume rendering techniques[40] assist scientists with viewing the characteristics of an entire flow field.

4.2 The Data Model

A rich and flexible data model is the central component of the visualization framework. The data model serves to organize the information in the complex, heterogeneous data sets obtained from multidisciplinary simulations. The object-oriented model represents the data in a manner similar to how the user views the data and their relationships. As a result, the model is knowledge-based rather than structure-based. That is, the model attempts to describe the knowledge content of the data by defining it semantically rather than reducing it to a collection of numerical data structures.

The core concepts of the object-oriented data model include [19]:

- *Object and object identifier:* An object represents a real world entity. The object identifier stores the name of an object and is system-wide unique.

- *Attributes and Methods:* The values of the attributes of an object constitute the state of the object. The set of methods associated with an object define behavior, operating on the state of the object.

- *Classes:* Classes organize collections of objects with similar attributes and methods. Objects are instances of a class and an object belongs to only one class.
- *Class Hierarchy and Inheritance:* A class may have any number of subclasses that inherit properties from this superclass. The concept of class hierarchy and inheritance of attributes and methods along the class hierarchy is what distinguishes object-oriented programming from programming with abstract data types.

The object-oriented model possesses several qualities that make it appropriate for managing scientific data. The capability to derive data by applying a method to a data object is an appealing aspect of the object-oriented model. This allows the designer to express the information content of the data without having to explicitly store all of its components. When the user queries a derived data variable, the system computes the values. If the derived data is queried frequently, the system may chose to permanently store the data.

The ability to extend the data model is also beneficial. The multidisciplinary model will undoubtedly grow as the technology becomes more defined and as additional disciplines are added to the simulations. The object-oriented model is modular and components can be easily added to expand the data model. For example, if acoustic data is integrated into the data model, an object representing the acoustic data set would be added to the current class structure. Additional methods and attributes could be defined to reflect the specific properties of the new discipline.

4.2.1 Data Model - Domain Knowledge

The following description of the object-oriented data model is based on the domain knowledge of a multidisciplinary, fluids/structures simulation. This high level data model allows for the semantic description of the data involved in this application. It is different from traditional data-structures-based models in that it encodes the knowledge about the data. The data model has been implemented in the Superglue programming environment; a description of the contents of the data model is given below. Figure 1 shows a simplified pictorial representation of the data model.

Objects and Object Identifiers Each multidisciplinary dataset is an object, representing the full configuration of a simulated vehicle such as the HSCT. This object contains all of the information about the initial conditions of the simulation, such as the flight speed of the aircraft (the Mach number), the angle of attack and the given conditions of the airstream (i.e. temperature, density, etc.). The multidisciplinary object also contains the discipline-dependent objects that represent data from the different disciplines in the simulation.

The fluids and structures data are instance objects of the discipline-dependent class of objects. As a result, both datasets have similar representations. Their differences are only apparent in the lower layers of their composition hierarchy, where grid objects and data objects are encoded. This differences arise because fluid dynamics data typically is based on structured grids and structural dynamics data is based on unstructured grids. In addition, the physical quantities that are represented on these grids are different.

Both the fluids and structures objects contain a list of component objects that represent the real entities that comprise the vehicle (i.e. wing, fuselage, tail). The definition of these components is typically defined by the grids that comprise the vehicle. For example, in the fluids case, there are typically several grids that define the volume about the vehicle. This is to ensure that the regions of interest are densely sampled with points, so that critical information is captured. The sample points must be carefully placed since the number of points that can be used in the simulation is limited by computing power. Points are densely positioned in the boundary layer, located near the surface of the solid objects, where the flow complexity is of interest.

Each of the component objects is further decomposed into timestep objects. Multidisciplinary simulations are time-dependent in nature in order to evaluate the dynamic aspects of the vehicle. Each timestep object consists of the current time in the simulation, a grid object and a data object. The timestep granularity for each discipline is different, so the number of timestep objects may not be equal in data objects for different disciplines.

The grid object has the information about the physical structure of the object. For the fluids data set this information includes a structured grid that defines the volume about a section of the vehicle. Additional information includes the locations of missing data points. This is a consequence of the multigrid composition of the fluids volume, where overlapping points in the grids are ignored. The grid object for the structures data set represents an unstructured grid. The information contained in this object includes the physical-space locations and the connectivity of the points comprising the grid.

The data object contains the data variables that are calculated at each of the points defined in the grid object. For the fluids data set, this contains the values for momentum, density and energy. The structures data set includes variables such as stress and deformation. Each of the objects described in this hierarchy has its own unique name and identifier. This allows it to be accessed independently and information about the object may be queried by the user.

Attributes and Methods Attributes define the state and characteristics of the objects. In the case of this object-oriented model, the attributes define the characteristics of the data that contribute to its mapping to a visual representation. Information such as the name of the data variable, the dimensions of the data set, whether the data is a scalar or vector field, and how the data variable is mathematically related to other data variables are some of the attributes that describe the state of the object. For example, the fluids data set contains the variable momentum. Momentum can be characterized as a three dimensional vector field of a given dimension which represents the current state of the object. Attributes also describe the initial conditions of the simulation such as the Mach number and the angle of attack of the vehicle.

Methods are functions that change the state of the object or create a new object. Methods are simply small functions that can be written to incorporate more knowledge into the data model. Methods can be applied to objects to create new objects, as in the case of derived data. The computed variables stored in the data object have methods attached to them to derive a data quantity. These methods are typically mathematical relationships. For example, variables such as velocity and pressure can be derived mathematically from momentum and density.

Methods are used to access and perform queries on the data objects. For example, if the user is interested in viewing the minimum and maximum data values of the pressure on the wing of the HSCT, a method can be invoked to query this value. The user can also invoke a method to view the composition of the data hierarchy.

Visualizations are created and rendered by invoking methods on the data objects in the hierarchy. Methods typically have the quality of being polymorphic. That is, these methods can be applied to a wide range of data types. For example, a method exists that draws contour lines. This method can be applied to the density field that is defined on the wing of the fluids data set or it can be applied to the stress field that is defined on the tail section of the structures data set. This aspect of the object-oriented paradigm allows for modularity and code reuse.

Classes Classes define a grouping of objects possessing similar characteristics. Classes are important in managing data for two reasons[19]. First, classes capture the semantics of the data sets, organizing them by their natural components. This organization permits the data to be accessed easily and efficiently. Second, classes provide the basis for querying the contents of the data model. For example, if the user is interested in querying the contents of each momentum vector to see how the minimum/maximum values change with respect to time, this can be done by simply searching through all objects defined by the data object class that contain momentum. This is easier than searching through all of the files containing the data sets for each timestep.

The objects defined previously are specific instances of classes, representing the actual entity rather than the abstract notion of it. For example, the HSCT multidisciplinary data set was defined as a specific instance of the multidisciplinary class. The fluids and structures objects were each instances of discipline-dependent classes.

Class Hierarchy and Inheritance Class hierarchies represent the relationship between all of the classes in the object-oriented model. The primary classes in this hierarchy include: the multidisciplinary class, the discipline-dependent class, the timestep class, the grid class, and the data class. This hierarchy is also shown in Figure 1.

Inheritance is one of the most appealing aspects of the object-oriented paradigm. Inheritance allows the attributes and methods to be passed down through the classes. Class hierarchy and inheritance allow for the hierarchical decomposition of the complexity involved in the multidisciplinary data set. This is beneficial in that this information only has to be recorded once.

4.2.2 Data Model - Data Structures

The data model is structured around the application-specific components that exist in a multidisciplinary fluids/structures simulation. As a result, its form is not general because it contains information specific to the represented domains. However, underlying data structures that comprise the data objects are general, storing the data in an efficient manner. These primitives are provided in the Superglue programming environment to support the large data sets found in CFD and multidisciplinary simulations.

Superglue contains primitives to efficiently manage large data sets[16]. Chunks are contiguous blocks of untyped memory, memory-mapped directly to the program's address space. Not only does this avoid exhausting swap space, but it also avoids the initial delay in file reading. The combination of efficient memory storage and the use of object classes and corresponding methods is a powerful approach to handling large data sets.

4.3 The User Model

To complement the data model in this visualization framework, the creation of a user model serves to describe the user and his characteristics. A user model is the knowledge source in a system that contains explicit assumptions on all aspects of the user that may be relevant to the behavior of the system. Although this aspect of the framework will not be directly studied in this project, a brief description of its role in the framework is discussed. This is an area of interesting research for future projects.

The information in the user model may include the interpretation aims of the user, the background of the user, and any limitations the user may possess. Interpretation aims (or "operations", as defined by Wehrend[41]) might include the following tasks: 1) identifying characteristics of the data such as points on a wing where the velocity field is zero, 2) locating areas of interest in the data such as low pressure regions, or 3) correlating data variables to see how the pressure from a fluid affects the structural integrity of a wing. Interpretation aims establish the goal of the user and the information he is trying to extract from the data. User background might include information such as the subjective meaning of certain color tables or shades as related to quantitative information, or a preference for certain visualizations and interaction techniques. Limitations of the user may include a color blindness, difficulties with psychomotor skills (*i.e.* working with a mouse), etc.

The user model can be represented using a stereotype hierarchy, as defined by Kass and Finin[18]. A simplified pictorial description of this user model is shown in Figure 2. This model uses stereotypes to describe a general class of users. Stereotypes define facts and rules that pertain to the user group's background, preferences and data analysis goals. Models are typically crafted for each application domain, usually by the explicit coding of domain-related goals, plans or knowledge that system users are expected to have. For this application, the hierarchy would be composed of two main stereotypes, namely the fluid dynamicists and the structural dynamicists. Each class of users has a specific academic background which has, in turn, led to the use of certain types of physical models, software, computing environments, etc.

These facts and rules can be of two types, either "definite" or "default". Definite facts and rules always apply to the given class of users. For example, fluid dynamicists always use mathematical relationships such as the Navier-Stokes equations to describe the relationships between data variables. Structural dynamicists use mathematical relationships that describe the deformation and stress of a body when a force or pressure is applied. Definite facts cannot be overridden, but information can be added to make them more precise.

Default facts and rules can be overridden as the stereotype hierarchy defines smaller classes of users. For example, a default rule might be that all fluid dynamicists use structured grids for their computations. However, this can be overridden if a certain group of fluid dynamicists uses unstructured grids in their work. The stereotype hierarchy can be further decomposed into smaller user classes and eventually to single users, who are represented as leaf nodes in this hierarchy.

The simplified user model for this project will incorporate the preferences and interpretation aims of a single scientist. The preferences of this scientist will be incorporated into the design of the visualization system. The interpretation aims of the scientist will change frequently as the scientist investigates different aspects of his data. As a result, interpretation aims will be specified by the scientist during a query to the visualization system. Three categories of interpretation aims have been chosen for incorporation into the prototype system. These interpretation aims, or goals, were obtained from Wehrend[41]: identify/locate, categorize/compare, and associate/correlate. The first category, identify/locate, deals with data of a single data variable in which single values are of interest (i.e. searching for the locations of where velocity values are equal to zero). The second category, categorize/compare, deals with multiple values, or a range of values, of a single data variable (i.e. locating low pressure regions by using contours that categorize the data). The third category, associate/correlate, deals with relationships between multiple data variables (i.e. correlating the data to see how pressure affects deformation). These three categories accommodate three types of interpretation aims that are of interest to the scientist. These interpretation aims may be redefined and/or expanded as the project evolves.

4.4 The Machine Model

The machine model is perhaps the most straightforward of the components comprising the visualization framework. This model details the capabilities of the computing environment available for scientific use, both in terms of software and hardware resources. The scientist should not be expected to know what type of environment he has available to him. This information is important.

however, impacting what type of visualization will be appropriate for a given configuration. Information will be encoded into the machine model, specifying the type of system, monitor resolution, graphics capabilities, color facilities, CPU, memory, available graphics libraries, etc. This model will describe a wide spectrum of machines, from laser printers (for publication) to workstations to virtual reality systems. The incorporation of the machine model makes the system very portable, allowing the scientist to view his data in many different environments.

This project will incorporate a simplified machine model, describing the current hardware and software used by the scientist. The hardware characteristics of this model reflect a Silicon Graphics (SGI) workstation. The capabilities of the SGI hardware and software will be incorporated when transforming the data into visualizations. The software characteristics reflect the capabilities of the Superglue programming environment and the visualization techniques that it possesses.

4.5 The Knowledge Base

The knowledge base contains information that will assist in the production of effective visualizations. This information includes rules and facts from graphic design and visual perception and the advice and knowledge of scientists in the different scientific domains. Rules and facts from graphic design have been documented by such researchers as Bertin[2] and Tufte[38]. Perceptual rules and guidelines are also readily available[5, 17]. Information has also been collected by researchers specifically for scientific visualization, including Senay and Ignatius[31] and Wehrend and Lewis [41]. Input from the scientific domains will be obtained by interviewing scientists with regard to which visualizations are most effective for them in the interpretation of data.

An "evaluation matrix" [8] will serve as the mechanism that will integrate the information in the knowledge base with that of the data, user, and machine models. The evaluation matrix will be three dimensional, having axes that represent data model characteristics, user model characteristics, and machine characteristics. Each location in the matrix will have one or more visualization techniques that fit the characteristics of the three dimensions. The evaluation matrix will be filled based on the rules of perception, graphic presentation and knowledge from the scientific domains. In order to fit these rules with the characteristics in the three models, the expressiveness and effectiveness criteria developed by Mackinlay[22] will be used. Expressiveness criteria determine how well a graphical representation expresses the information stored in the data. Effectiveness criteria determine if the graphical representation takes advantage of the output medium and the human visual system. If more than one visualization exists, they will be prioritized according to this evaluation process.

For example, the axis describing the data characteristics will contain coordinates representing the contents of a leaf node on the data model (i.e. "fluids data, wing object, timestep 1, 3 dimensional vector field, physical variable is velocity"). In order to represent relationships between data variables, all combinations of data variables will also be encoded. The axis describing the user will be similar, containing the characteristics of an individual user (i.e. "Fluid dynamicist, NASA Ames Research Center, research in particle simulation, prefers point-and-click interface, color blind"). Finally, the axis describing the machine model will contain details about the specific hardware available (i.e. "Silicon Graphics 420/VGX, true color, 48 MB RAM"). The visualization technique that would support these three factors would be that of a particle trace or flow ribbon. This is derived from the following rule, attributed to Shirley and Neeman[34] and compiled by Senay and Ignatius [31]: "Particle advection does not necessarily show twisting in a vector field. If it is desirable to display twisting effects, one should use flow ribbons." A pictorial description of the evaluation matrix is shown in Figure 3.

The evaluation matrix for this project will be an $N \times 1 \times 1$ matrix, where N is the number of data variables and data variable combinations obtained from the data model. Since only one user and one machine will be used for this project, the dimensions of the two remaining axes will be one. At first these mappings will be performed by hand and encoded into the evaluation matrix. Future research would involve automating the generation of the evaluation matrix based on the information in the models and the rules in the knowledge base.

4.6 The Prototype Visualization System

This prototype system will be implemented in Superglue[16], an object-oriented programming environment based on the language Scheme. Superglue is under development at the NASA Ames Research Center and supports a rich base of visualization primitives. As a result, the programming of visualization techniques can be kept to a minimum by taking advantage of previously written software. The system will use the knowledge embodied in the data model and the knowledge base. A simplified user and machine model will be incorporated into the creation of the mappings stored in the evaluation matrix.

A prototype interface has been developed to lead the scientists through the visualization process. These interface pictures are preliminary and they will be presented to the scientist and iterated upon during the course of the project. The interface will ask the scientist to input what data set they would like to analyze. Figure 4 shows this main interface with four data sets the scientist may select from. The interface will then ask for input regarding what data items the scientist is interested in. Figure 5 shows the template that the scientist will use to select data. The scientist must also select the interpretation aim (or "goal" as specified in the interface) that he wishes to perform on the selected data variables. The system will suggest and display the visual representations based on this data query and the information modeled in the framework, specifically, the data model. The results from the separate visualizations will be displayed separately or merged together if the representations do not detract from each other. If the results are displayed separately, the visualizations will share the same orientation. Figure 6 shows the resulting visualizations.

Interactivity and direct manipulation will be an important aspect of this prototype visualization system. The system will allow the user to manipulate the resulting image. Multiple, coordinated views give the scientist the ability to associate the data sets by coordinating the viewing perspective in each window.

If the scientist is not satisfied with the selection made by the system, they have the option to override and select their own visualization. This can be done by selecting the "User Select" button and choosing which visualization technique they are interested in. This is shown in Figure 7. The system will determine if the selected visualization can be applied to the selected data and present the result to the scientist.

In order to illustrate how the scientist would interact with the data-model-based system, the following brief scenario is presented. A structural dynamicist is interested in viewing how the forces of the fluid flowing over the vehicle affect the structural integrity of the wing on the HSCT. In order to view this relationship, he first selects the HSCT data set (Figure 4). He is then required to input which object he is interested in viewing, the range of timesteps, the data variables that influence the phenomenon, and his intepretation aim (Figure 5). He selects the wing object and the last 10 timesteps. The physical variables that he is interested in include the pressure of the fluids data set and the deformation of the structures data set. His interpretation aim is to correlate the data variables so that he may understand how one variable affects the other. The scientist is presented with several visualizations depicting the query that he has made (Figure 6).

The first visualization shows the distribution of pressure over the wing using contours. This visualization has a cross-bar cutting slices in the wing and showing a plot of the pressure distribution at this slice. Plots are helpful to the scientists in performing a more quantitative analysis of the data.

They should be used often in conjunction with qualitative visualizations. The third representation views the deformation in animated form, cycling through the timesteps. Animation is effective in viewing how the deformation warps the object over time. Since all representations are coordinated, each representation is cycling together with the deformation. The scientist is advised that if he is interested in stopping and investigating a certain timestep, that he may select a timestep using the slider bar provided in the window. The fourth representation also corresponds to the cross bar in the first visualization (it is also presented in the animation) allowing the viewer to see the plot of the deformation across the cross section of the wing. As mentioned previously, the viewpoints of all representations are coordinated in the windows. The viewpoint can be changed in any of the visualizations and this change will be reflected in the other representations. If the scientist is interested in viewing visualizations disjointly, he has the option of "disconnecting" one or all of the visualizations.

The data model is responsible for interacting with the knowledge base in generating effective visualizations. However, the scientist should also be able to access the knowledge in the data model in performing queries about the contents of the data set. The data model should be clearly represented to the scientist so he may interact with it easily. This is accomplished using a graphical interface that allows the scientist to investigate the data model and browse through its components. The graphical representation of the data model presents information to the scientist about the contents of the data. This information includes the basic attributes of the data such as the size and dimensionality of the data sets. A sample interface that demonstrates how the scientist may query the data model is presented in Figure 8. This picture of a prototype interface shows the scientist querying the object-oriented model and viewing the components of an object in the data model.

One of the benefits of incorporating all of the semantic information about the data in the model is that the system can automatically annotate the visualization(s) using the semantic information stored in the data model. For example, if the scientist queries the pressure on the wing, the resulting visualization might present a contour map with the information about the pressure values labeled as shown in Figure 6. This could include information about the specified data object: initial condition information, the range of pressure values, a labeled color table, etc. This information is often ignored and this can lead to misinterpretation of the data[12].

The current version of the software that implements this framework consists of the base layer of the data model. Work is in progress to integrate the data model with the visualization primitives that are available in the Superglue programming environment. The user is currently able to view the contents of the multidisciplinary data model and to query its contents. The only visualization technique available is the representation of the grid geometries.

4.7 User Interaction and Response

The scientist will be a part of the entire design process of this system. This will allow for interaction between the scientist and the designer regarding the development of the system. Participation by the scientist in the design process will help to assure that the system fits the needs and interests of the user.

Scientists will tested using the system during its development cycle and upon completion. The goal of this interaction is to assess the effectiveness of this type of approach in the development of visualization systems. User testing will involve the "Thinking Aloud" approach[20], which encourages scientists to verbally express their thoughts while working with the system. This is effective in pointing out difficulties in using the system. This cognitive technique has been successful in the past in determining problem areas in the evaluation of scientific software[23].

The user will also be surveyed and asked a series of questions concerning past experiences with visualization systems and questions about the current approach. Some of the issues that will be addressed include ease-of-use, productivity, flexibility, and trust in the accuracy of the generated visualizations.

Response from the user will be the only real way to know if the use of assistant-based systems is a plausible approach. User input is critical throughout the design project to determine if such an approach to visualization systems is a positive step.

5 Contributions of Proposed Work

The contribution of this work comes in two parts, a theoretical foundation and a user tested implementation. In the design of the assistant-based system, the development of a structured framework is required in order to define the necessary components that must comprise such a system. The development of this framework provides a structure from which systems can be built and expanded. This structured approach to visualization, as opposed to current *ad hoc* approaches, is a step towards understanding the visualization process as a whole.

The development of an assistant-based visualization system will help create effective visualizations for the scientist. This leads to more accurate interpretation of the data since knowledge from various domains such as graphic design and visual perception, goes into the development of the representation. This knowledge is typically not possessed by the scientist who often relies on the help of a visualization expert. By freeing the scientist from having to consult with this expert, the visualization process can be made more productive. This approach is a significant contribution to the design of visualization systems in that it makes visualization technology more accessible to the typical scientist.

Although work has been done in encoding information about graphical representations, this project will attempt to codify this information in such as way that it can be applied to visualization systems. The research presented in this proposal builds off the foundation set by the previous research projects mentioned in Section 3.2. It expands on this research by modeling the all of the components that comprise the visualization process. As a result, the necessary information for generating effective graphics is available. Differences between previous research and this project are described below.

Haber and McNabb's work does not implement data, user or machine models to formalize the visualization process. However, they place an emphasis on the generation of the visualization mappings that are essential in creating visualizations. Mackinlay's APT was intended for graphical presentations of relational information. Mackinlay's designs were not oriented towards scientific visualization, but rather the development of charts and graphs. Robertson and De Ferrari have acknowledged the need for a formalization of the visualization process, but have not gone beyond defining the need for data models and the user's goal-related input. They currently do not have an implemented system. Wehrend and Lewis's work is a categorization of visualization techniques based on the characteristics of the data and the perceptual task to performed on the data. They did not attempt to encode this knowledge in the development of a visualization system. Senay and Ignatius do not define the components specified in this proposed framework, specifically, the data model. However, they examine how the user and the task at hand are involved in generating visual representations[33] and have an extensive knowledge base of perceptual and graphic design rules. Senay and Ignatius do have a system that is currently implemented, although the design time for creating a visualization is prohibitive.

This project builds on the individual strengths of these research projects. In addition, this project creates the necessary framework for implementing an assistant-based visualization system.

Specifically, a great deal of attention will be paid to defining the data model and using the knowledge contained in the data to generate visualizations. Because the knowledge will be encoded in the evaluation matrix, the generation of visualizations will only require a table lookup. Therefore, there will be almost no delay in suggesting the visualization in addition to the rendering time.

By incorporating a data model, this work will emphasize data management and representation issues that have been missing from most visualization systems. The data model will incorporate knowledge about the data as opposed to defining data structures which merely store the data. This will allow the user to view the data in their own terms and not in the form of data structures defined by computer scientists. It will also allow the system to develop effective visualizations based on this knowledge. Current systems lack data models which contain the high level knowledge that can assist the scientist with their data analysis. The object-oriented paradigm will be used as the basis for the data model. The object-oriented paradigm allows for modularity, code reuse, expandability and the capability to derive data usings methods.

This solution addresses the challenge of organizing and managing multiple, heterogeneous data sets and their resulting visual representations. Since multidisciplinary simulations will be more common as computing technology advances, this type of system will explore data management issues. Data management has finally been recognized by the visualization community as an essential component of any visualization system. This project attempts to address this issue by incorporating object-oriented data modeling techniques.

A disadvantage of this type of system is that it will lack the flexibility that some scientists would like. If this is the case, the user may always select her own visualization (as shown in Figure 7).

There will be many difficulties in the development of this project. The most formidable problem will be in gathering and representing the vast amount of knowledge from the scientific domains into a knowledge base that can be accessed and used effectively. Another problem will be creating the interface with which the scientist can deal with his data in the most natural and intuitive way. Many problems will undoubtedly be posed by the large, heterogeneous data sets.

6 Timeline

In order to map out how this project will evolve, a sequence of milestones is presented. This will be useful in keeping the project on course and assessing progress. In addition, it attempts to define when this project will be considered "done". Figure 9 shows how the following work segments are distributed over the course of the project.

A. *Encode the Data Model:* Create the base structure of data model, incorporating all of the data variables that are involved in the fluid/structure interaction scenario. Encode this information so that it fits into the Superglue programming environment. This will ensure that the software takes advantage of what Superglue has to offer and does not stray from the main goal of the research. Encode information about the data attributes, constraints on the data, relationships between data variables, including derived data types.

B. *Unstructured Grids:* Some programming will be required to incorporate visualization techniques for unstructured grids so that structural dynamics data sets can be visualized. Members of the Superglue development team will assist with this development to further the capabilities of their environment.

C. *Visualization Support:* Ensure that the visualization system can provide support for all of the visualization techniques that are incorporated in the system. Provide for facilities to label data, provide color tables, etc. so that information is well annotated and the scientist can easily understand what their visualizations mean.

D. *User/Machine Models:* Set up a simple user and machine model that will contain information about a single user and a single machine in the scientist's environment.

E. *Perception/Graphic Design Research:* Gather information from visual perception and graphic design from the various sources that have researched this issue.

F. *Work with Scientist:* Perform a study with a scientist as to which representations best represent his data. Spend time with the scientist learning about his/ working environment, habits, and extracting knowledge that can be embodied for the generation of visualizations.

G. *Encode Knowledge:* Based on the information gained from the scientist, the rules from graphic design and perception, and the description of the data, encode appropriate visualization techniques into the evaluation matrix using the evaluation criteria. Together with this encoding, provide the rules and reasons behind the selection of visualization techniques. Provide for the use of multiple visualization techniques for a single query and determine how they will be prioritized.

H. *User Interface:* Develop a user interface that will provide easy access to the data and in the specification of the goals. Provide for messaging window which states why selected visualizations were chosen.

I. *User Survey:* Survey the scientist to determine the effectiveness of this type of approach.

J. *Write Up Results:* Perform final analysis and write up results in final draft of the dissertation.

7 Conclusions and Future Research

In order to effectively analyze the vast amounts of data being generated today by complex computer simulations, scientists require the help of some type of intelligent assistance. Current visualization systems are often overpowering and do not provide the scientist with much guidance as to how to proceed with their data analysis. Without this assistance, the scientist is required to learn the complexities of generating visualizations and understand how to use visualization software packages. This knowledge base is broad, requiring expertise in visualization, graphical presentation, visual perception and knowledge of the application domains.

The principal contribution of this work is the development of an assistant-based visualization system, based on a structured framework, for the design of effective graphics. The components of this framework include: a data model, a user model, a machine model, and a knowledge base. Each of these components defines its domain and contributes to the development of useful visualizations. The emphasis of this project will be the development of the data model, the central component of this framework.

This approach to visualization is intended to reduce the current level of complexity in visualization systems, transforming them from "systems the scientist must serve into productive tools that serve the scientist"[6]. The end result will lead to increased productivity for scientists so that they can more efficiently analyze the large data sets that are produced by modern simulation technology.

Future research plans include incorporating full user and machine models that take into account the task at hand as well as the varying types of input and output devices available to scientists. This effort will complete the characterization of the visualization environment and demonstrate the impact that each of these components has in the development of visualization systems. Automating the application of the knowledge base rules to the individual models would be another major contribution. Currently the rules are applied by hand. If this process could be automated, the time required to generate the evaluation matrix would be substantially reduced. Finally, due to the large size of the data sets that will be encountered, database-oriented issues will need to be addressed. A distributed, persistent object database derived from the original data model would be valuable for managing large multidisciplinary data sets.

Acknowledgements

I would like to thank Gitta Domik (University of Colorado), Sam Uselton, and Jeff Hultquist for their help and guidance in the development of this proposal. Thanks also to Al Globus for pointing out important issues concerning the data model and to John Dannenhoffer (United Technologies Research Center) for thought-provoking discussion.

References

[1] Gordon V. Bancroft, Fergus J. Merritt, Todd C. Plessel, Paul G. Kelaita, R. Kevin McCabe, and Al Globus. FAST: A Multi-Processed Environment for Visualization of Computational Fluid Dynamics. In Arie Kaufman, editor, *Proceedings of Visualization '90*, pages 14–24. IEEE Computer Society, 1990.

[2] J. Bertin. *Semiology of Graphics.* University of Wisconsin Press, Madison, WI, 1983.

[3] Grady Booch. *Object Oriented Design with Applications.* The Benjamin/Cummings Publishing Company, Redwood City, CA, 1991.

[4] Steven M. Casner. A Task Analytic Approach to the Automated Design of Graphic Presentations. *ACM Transactions on Graphics*, 10(2):111–151, April 1991.

[5] W.S. Cleveland and R. McGil. Graphical Perception: Theory, Experimentation, and Applications to the Development of Graphical Methods. *Science*, pages 828–833, August 1985.

[6] Michael Dertouzos. Modern Interfaces for Ancient Humans. Presentation at the MIT Media Lab Symposium on Interface Agents, October 20, 1992.

[7] Gitta Domik. The Role of Visualization in Understanding Data. In *Lecture Notes on Computer Science 555: New Trends and Results in Computer Science*, pages 91–107. Springer Verlag, 1991.

[8] Gitta Domik. Toward a Formal Description of Visualization for Earth and Space Sciences. Proposal submitted to NASA, Innovative Research Program, OSSA, 1991.

[9] Gitta Domik. A Paradigm for Visual Representations. Technical report, University of Colorado, Computer Science Department, CB 430, Boulder, CO 80309-0430, 1992. In preparation.

[10] Steven Feiner, Jock Mackinlay, and Joe Marks. Automating the Design of Effective Graphics for Intelligent User Interfaces. In *Tutorial at the 1992 Conference on Computer Human Interaction*, 1992.

[11] Al Globus, Creon Levit, and Tom Lasinski. A Tool for Visualizing the Topology of Three-Dimensional Vector Fields. In Gregory M. Nielson and Larry Rosenblum, editors, *Proceedings of Visualization '91*, pages 33–40. IEEE Computer Society, 1991.

[12] Al Globus and Eric Raible. Thirteen Ways to Say Nothing with Scientific Visualization. Technical Report RNR-92-006, NASA Ames Research Center, MS T045-1, Moffett Field, CA 94035, February 1992.

[13] Robert B. Haber and David A. McNabb. Visualization Idioms: A Conceptual Model for Scientific Visualization Systems. In G.M. Nielson and B. Shriver, editors, *Visualization in Scientific Computing.* IEEE Computer Society Press, 1990.

[14] Richard Hull and Roger King. Semantic Database Modeling: Survey, Applications, and Research Issues. *ACM Computing Surveys*, 19(3), September 1987.

[15] J.P.M. Hultquist. Constructing Stream Surfaces in Steady 3D Vector Fields. In Arie E. Kaufman and Gregory M. Nielson, editors, *Proceedings of Visualization '92*, pages 171–178, 1992.

[16] J.P.M. Hultquist and E.L. Raible. Superglue: A Programming Environment for Scientific Visualization. In Arie E. Kaufman and Gregory M. Nielson, editors, *Proceedings of Visualization '92*, pages 243–251, 1992.

[17] D. Kahneman and A. Henik. Perceptual Organization and Attention. In M. Kubovy and J.R. Pomerantz, editors, *Perceptual Organization*, pages 181–211. Lawrence Erlbaum, 1981.

[18] Robert Kass and Tim Finin. General User Modeling: A Facility to Support Intelligent Interaction. In Joseph W. Sullivan and Sherman W. Tyler, editors, *Intelligent User Interfaces.* ACM Press, 1991.

[19] Won Kim. *Introduction to Object-Oriented Databases.* The MIT Press, 1990.

[20] Clayton Lewis. Using the "Thinking Aloud" Method in Cognitive Interface Design. Technical report, IBM Thomas J. Watson Research Center, Yorktown Heights, NY, 10598, 1982.

[21] Bruce Lucas, Gregory D. Abram, Nancy S. Collins, David A. Epstein, Donna L. Gresh, and Kevin P. McAuliffe. An Architecture for a Scientific Visualization System. In Arie E. Kaufman and Gregory M. Nielson, editors, *Proceedings of Visualization '92*, pages 107–114, 1992.

[22] Jock Mackinlay. Automating the Design of Graphical Presentations of Relational Information. *ACM Transactions on Graphics*, 5(2):110–141, April 1986.

[23] Kristina D. Mickus. Participatory User Interface Design for Scientific Visualization Systems. Master's thesis, University of Colorado, Boulder, 1991.

[24] National Center for Supercomputer Applications, Champaign, IL. *NCSA HDF Calling Interfaces and Utilities, Version 3.0*, 1989.

[25] William T. Olle. *The CODASYL Approach to Database Management.* Wiley, 1978.

[26] R.K. Rew and G.P. Davis. NetCDF: An Interface for Scientific Data Access. *Computer Graphics and Applications*, July 1990.

[27] Y.M. Rizk and K. Gee. Numerical Prediction of the Unsteady Flowfield Around the F-18 Aircraft at Large Incidence. In *29th AIAA Aerospace Sciences Meeting*, Reno, NV, January 1991. AIAA Paper No. 91-0020.

[28] Philip K. Robertson. A Methodology for Scientific Data Visualization: Choosing Representations Based on a Natural Scene Paradigm. In Arie Kaufman, editor, *Proceedings of Visualization '90*, pages 114–123. IEEE Computer Society, 1990.

[29] Philip K. Robertson. A Methodology for Choosing Data Representations. *IEEE Computer Graphics and Applications*, 11(3):56–67, May 1991.

[30] Philip K. Robertson, Lisa De Ferrari, Peter Fletcher, and Guy Vezina. Visualisation on a Massively Parallel Supercomputer. Technical Report TR-HJ-91-04, CSIRO Division of Information Technology, Center for Spatial Information Systems, GPO Box 664, Canberra, ACT 2601, Australia, 1991.

[31] Hikmet Senay and Eve Ignatius. Rules and Principles for Scientific Data Visualization. Technical Report GWU-IIST-90-13, George Washington University, Institute for Information Science and Technology, Department of Electrical Engineering and Computer Science, School of Engineering and Applied Science, Washington, D.C., 20052, May 1990.

[32] Hikmet Senay and Eve Ignatius. VISTA: Visualization Tool Assistant for Viewing Scientific Data. In *SIGGRAPH 1990 Course Notes: State of the Art in Data Visualization*, 1990.

[33] Hikmet Senay and Eve Ignatius. A Task-Oriented Approach to Scientific Visualization Using Perceptual Dimensions and Organization. In *IEEE Visualization '92 Workshop on Automated Design of Visualizations*, October 1992.

[34] P. Shirley and H. Neeman. Volume Visualization at the Center for Supercomputing Research and Development. In *Chapel Hill Volume Visualization Workshop*, May 1989.

[35] Silicon Graphics Corporation, Mountain View, CA. *Explorer User's Guide, Module Writer's Guide*, 1991.

[36] L.A. Treinish and M.L. Gough. A Software Package for Data Independent Management of Multidimensional Data. *EOS Transactions, American Geophysical Union*, 68, 1987.

[37] Lloyd A. Treinish. Panel on Grand Challenge Problems in Visualization Software. In Arie Kaufman and Gregory Nielson, editors, *Proceedings of Visualization '92*, pages 366–371. IEEE Computer Society, 1992.

[38] E. Tufte. *The Visual Display of Quantitative Information.* Graphics Press, Chesire, CT, 1983.

[39] Craig Upson. The Application Visualization System: A Computational Environment for Scientific Visualization. *IEEE Computer Graphics and Applications*, 9(4), July 1989.

[40] Samuel P. Uselton. Volume Rendering for Computational Fluid Dynamics: Initial Results. Technical Report RNR-91-026, NASA Ames Research Center, MS T045-1, Moffett Field, CA 94035, September 1991.

[41] Stephen Wehrend and Clayton Lewis. A Problem-Oriented Classification of Visualization Techniques. In Arie Kaufman, editor, *Proceedings of Visualization '90*, pages 139–143. IEEE Computer Society, 1990.

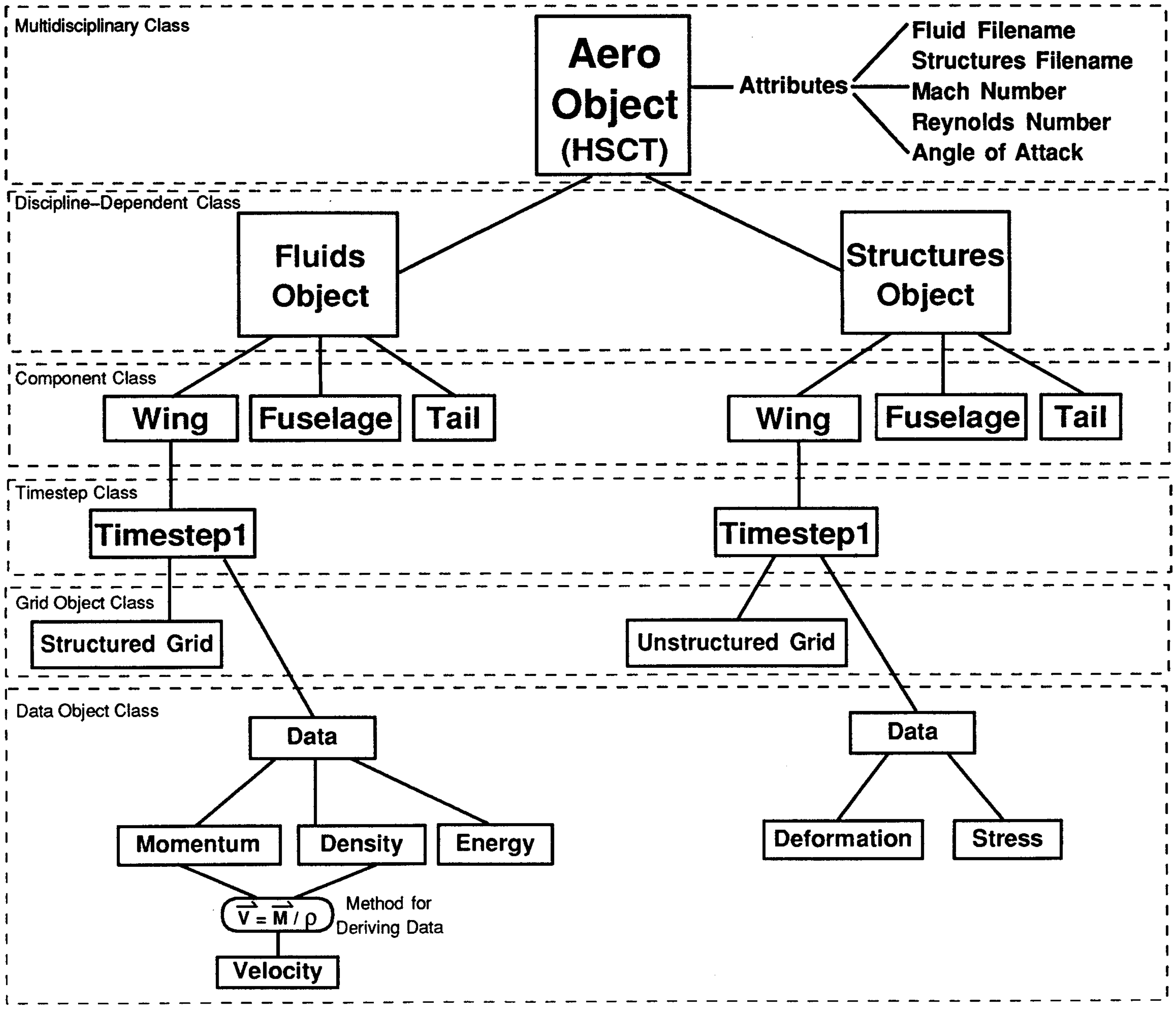


Figure 1: Fluids/Structures Data Model

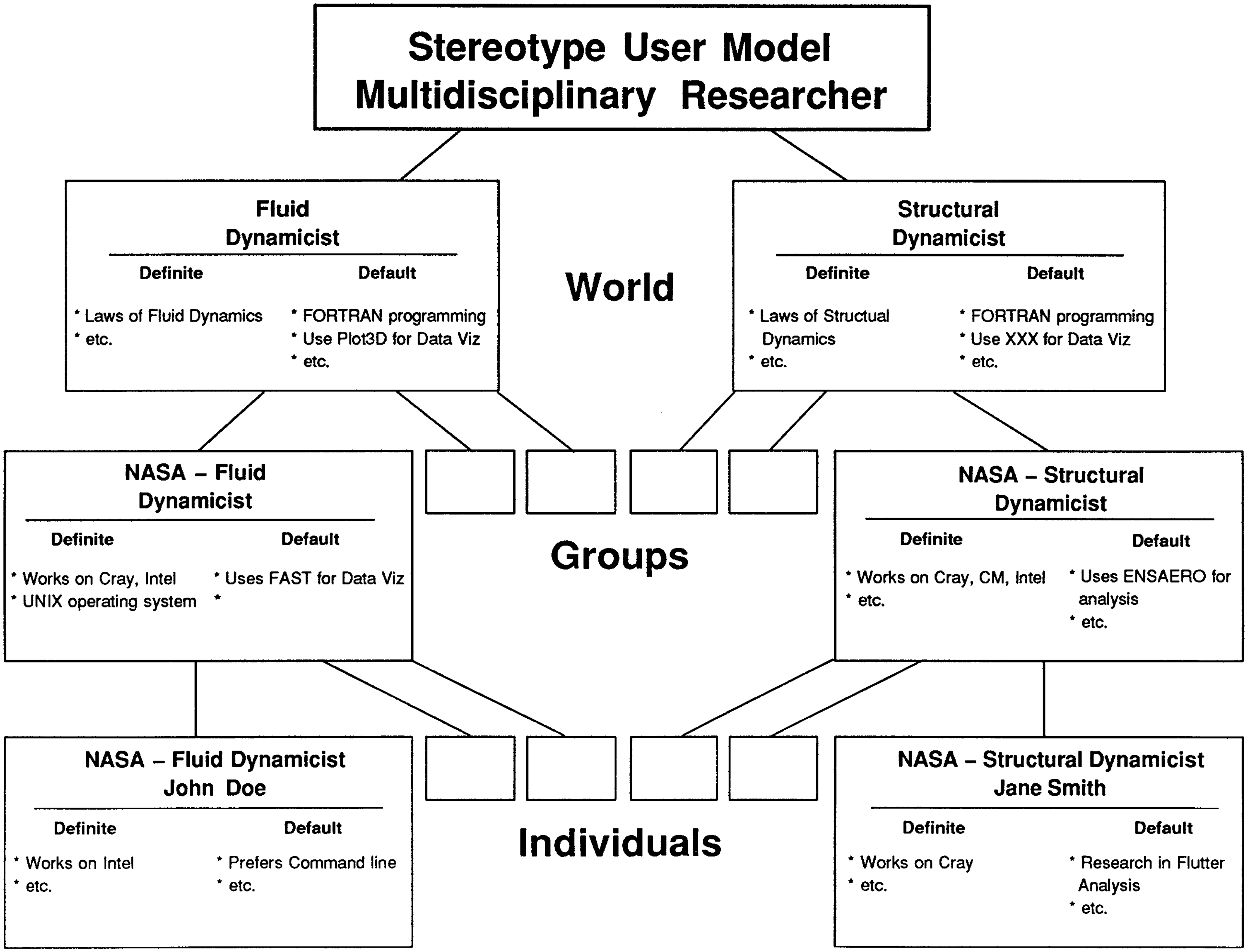


Figure 2: Fluids/Structures User Model

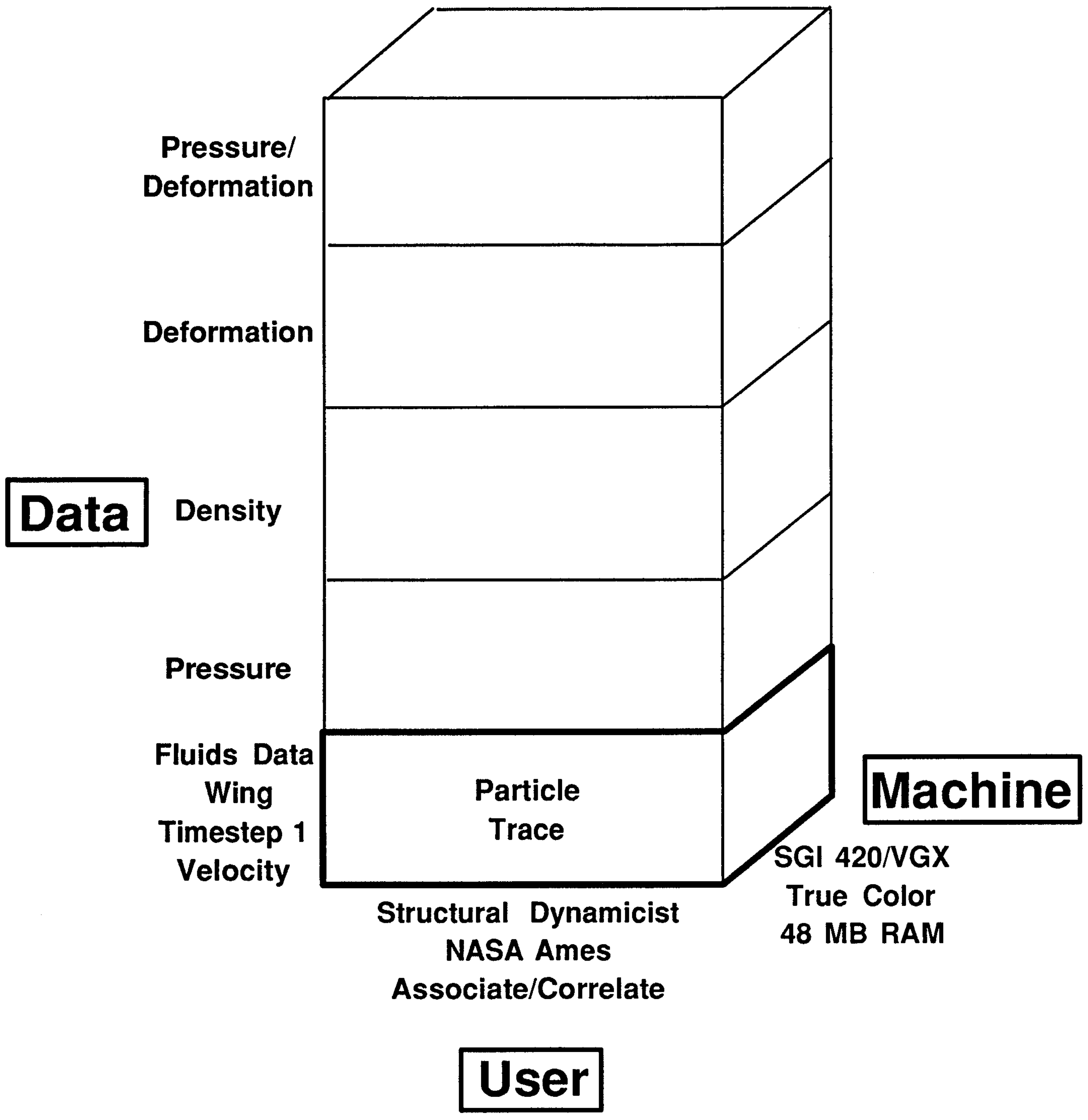
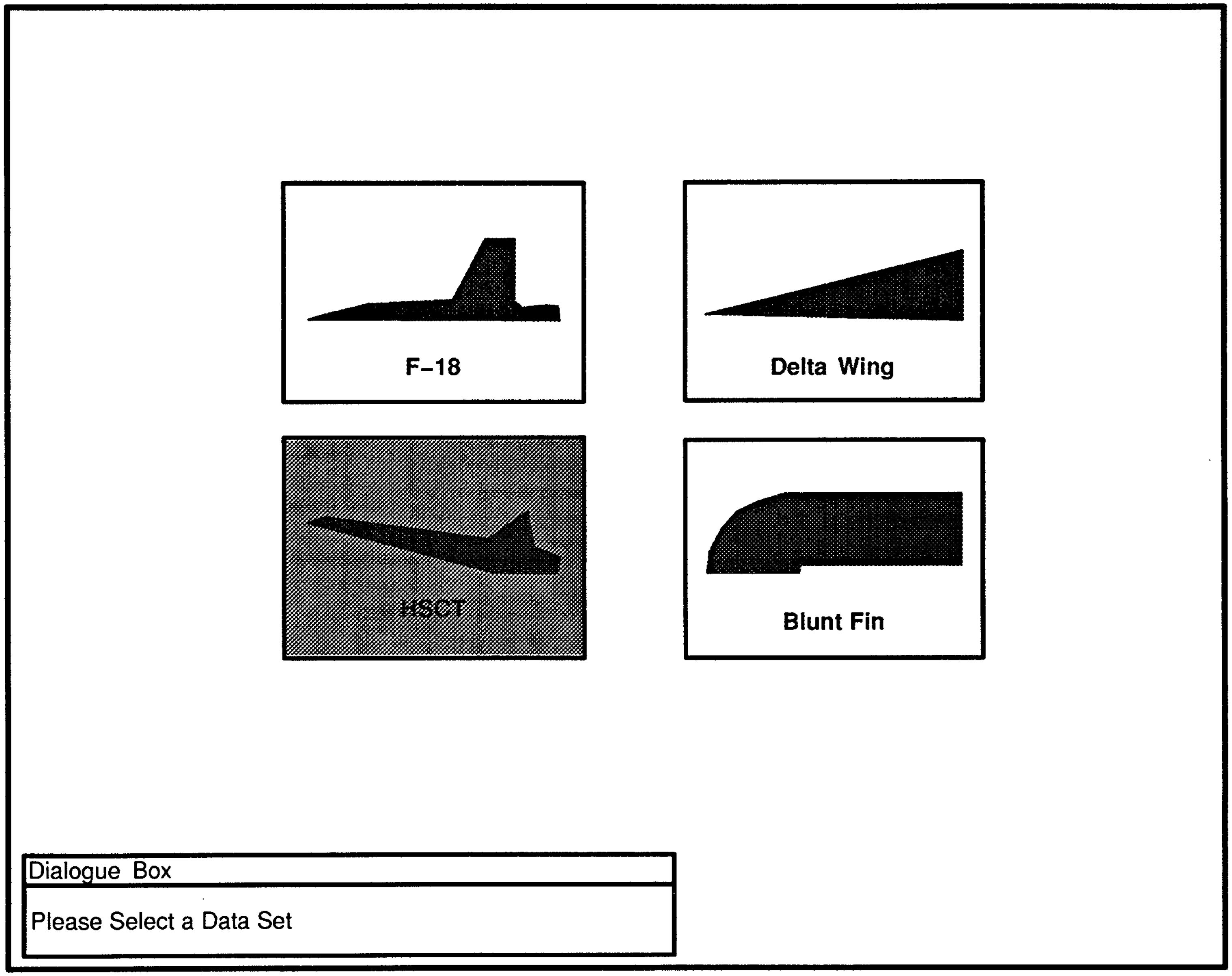
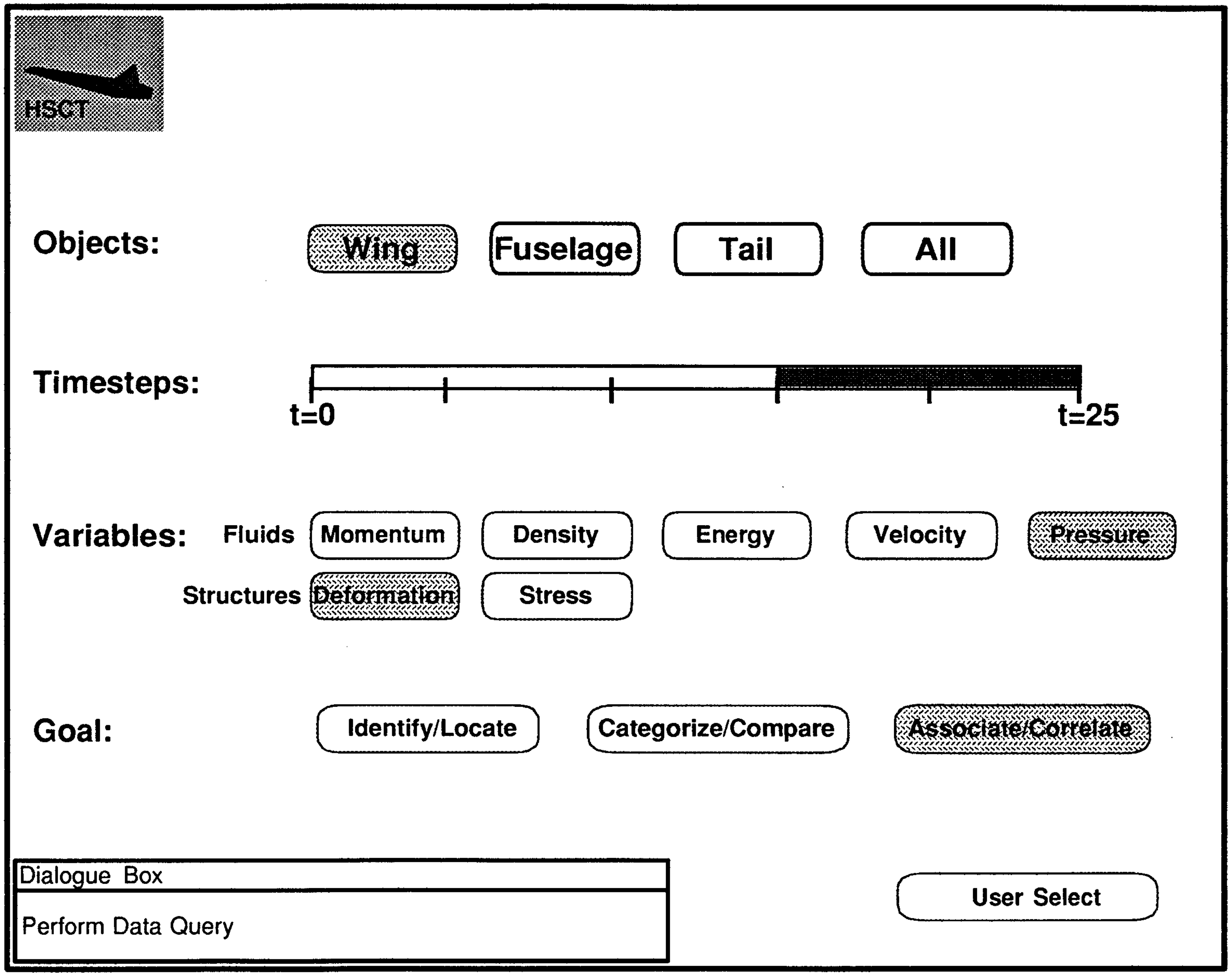


Figure 3: Evaluation Matrix



Please Select a Data Set

Figure 4: Prototype Interface: Main Interface



Objects: [✓ Wing] Fuselage Tail All

Timesteps: t=0 t=25

Variables: Fluids Momentum Density Energy Velocity [✓ Pressure]

Structures [✓ Deformation] Stress

Goal: Identify/Locate Categorize/Compare [✓ Associate/Correlate]

Dialogue Box

Perform Data Query

User Select

Figure 5: Prototype Interface: Data Input

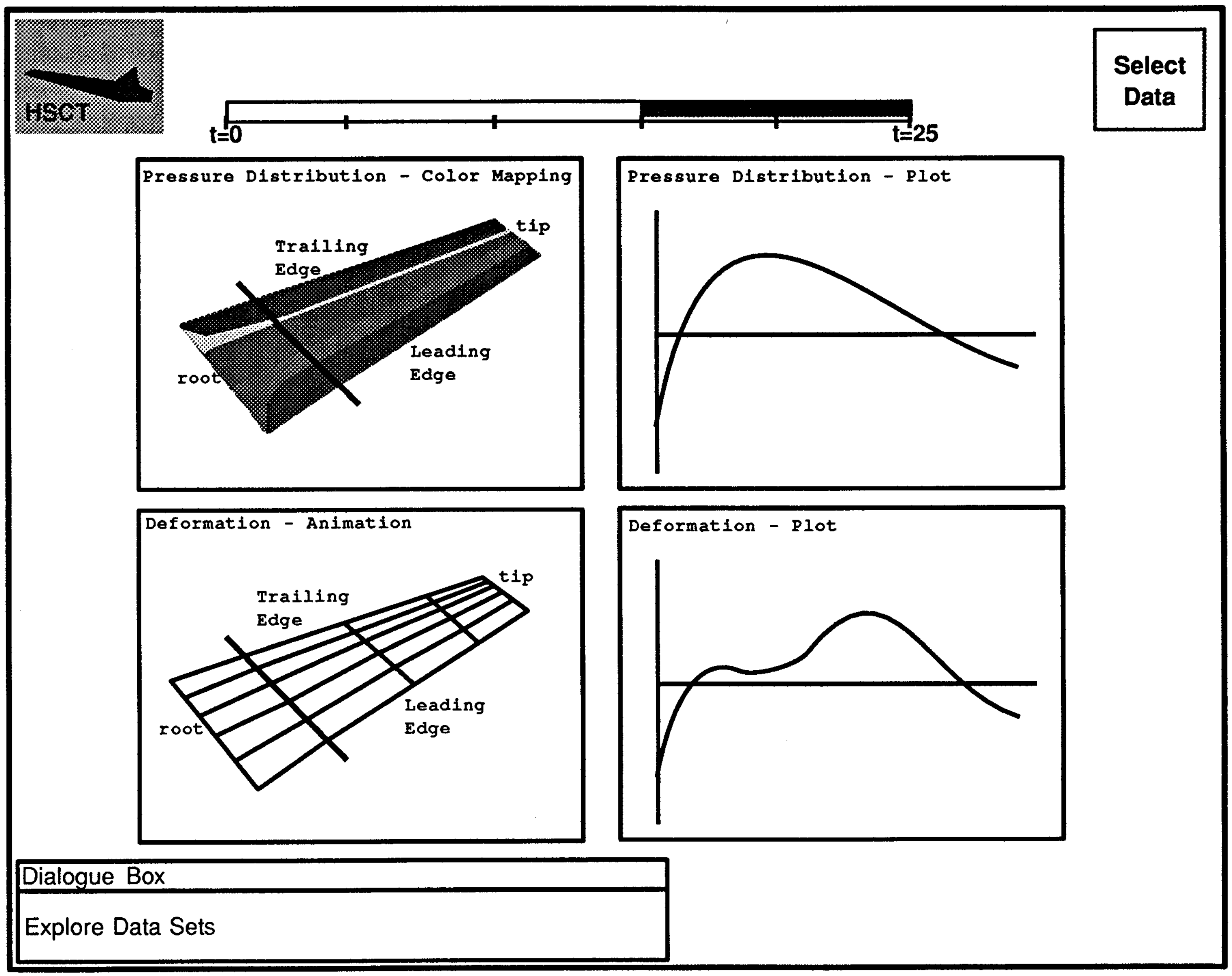
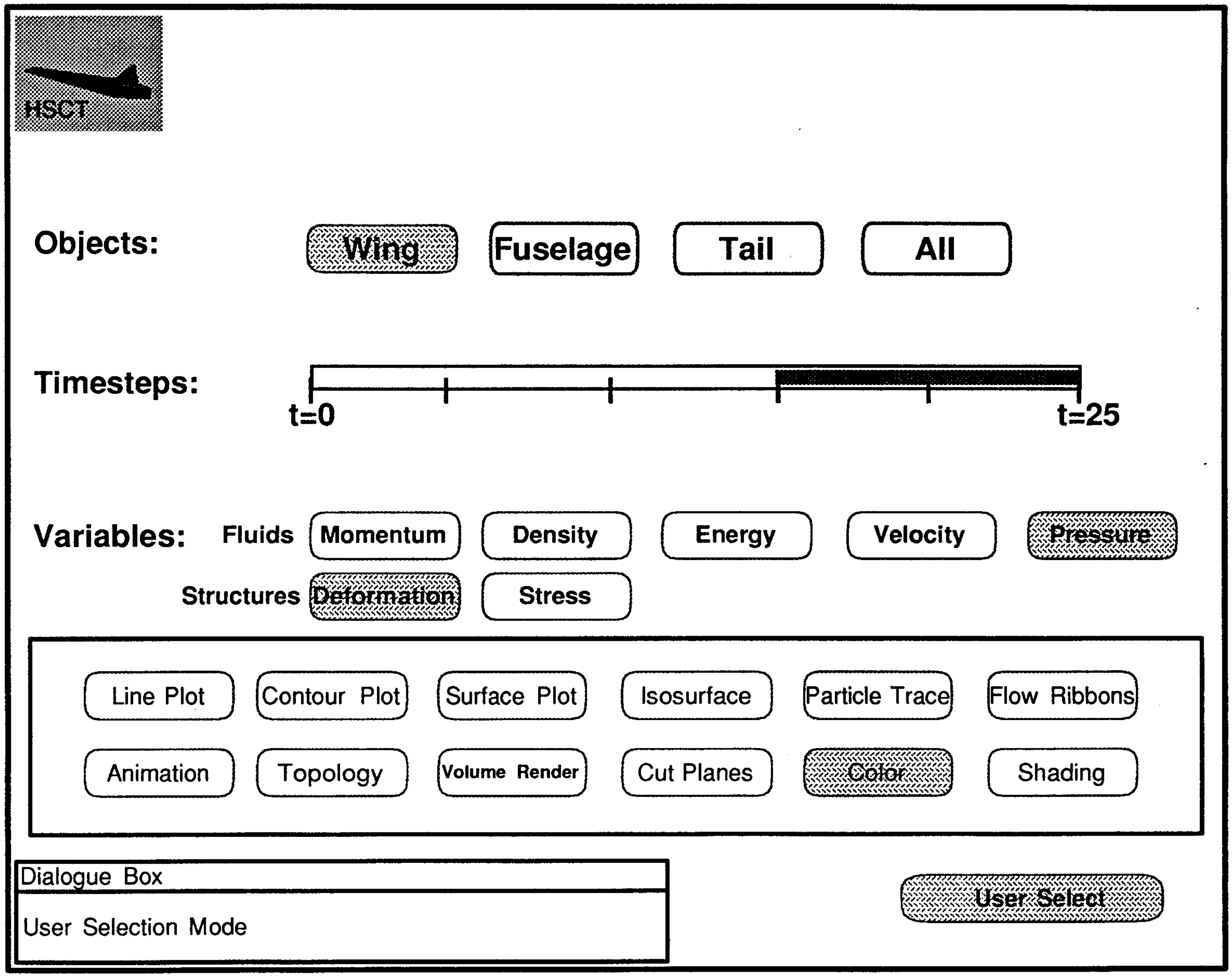


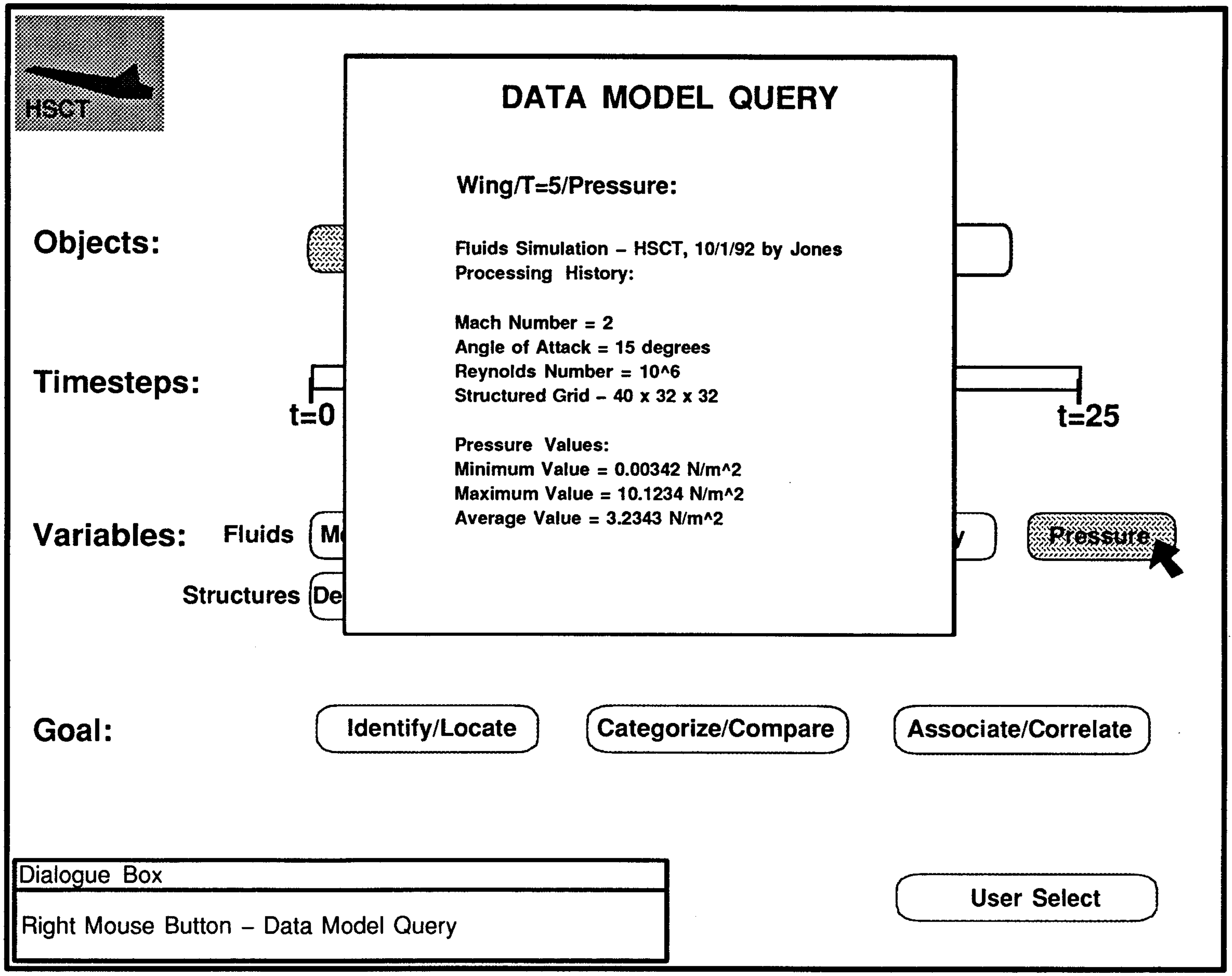
Figure 6: Prototype Interface: Suggested Visualizations



Dialogue Box

User Selection Mode

Figure 7: Prototype Interface: User Selection



DATA MODEL QUERY

Wing/T=5/Pressure:

Fluids Simulation – HSCT, 10/1/92 by Jones
Processing History:

Mach Number = 2
Angle of Attack = 15 degrees
Reynolds Number = 10^6
Structured Grid – 40 x 32 x 32

Pressure Values:
Minimum Value = 0.00342 N/m^2
Maximum Value = 10.1234 N/m^2
Average Value = 3.2343 N/m^2

Dialogue Box

Right Mouse Button – Data Model Query

Figure 8: Data Model Query

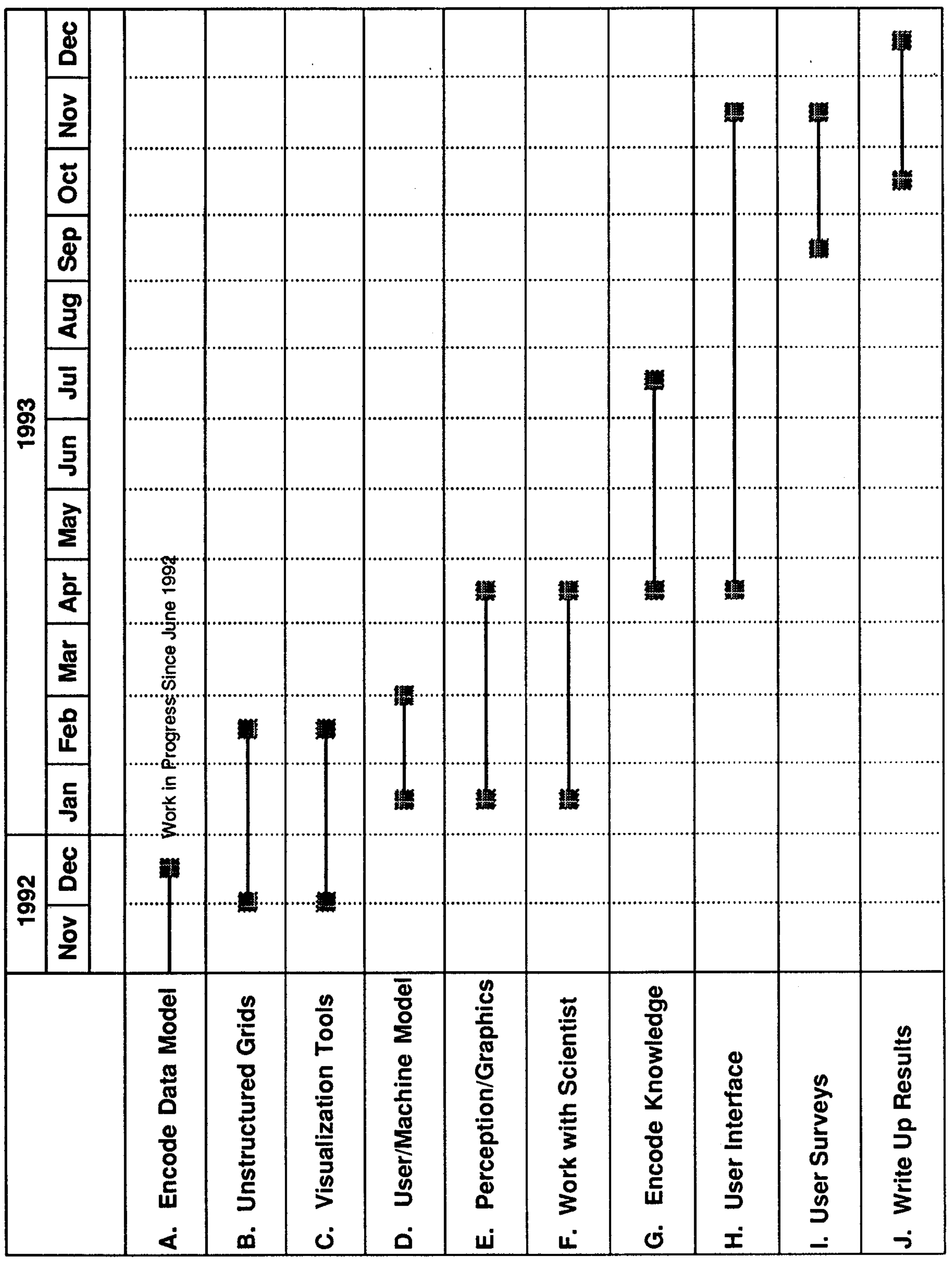


Figure 9: Research Timeline